



Tecnológico de Monterrey

M4. Tarea - Manejo de Tiempo en Unity

Manuel Villalpando Linares A01352033

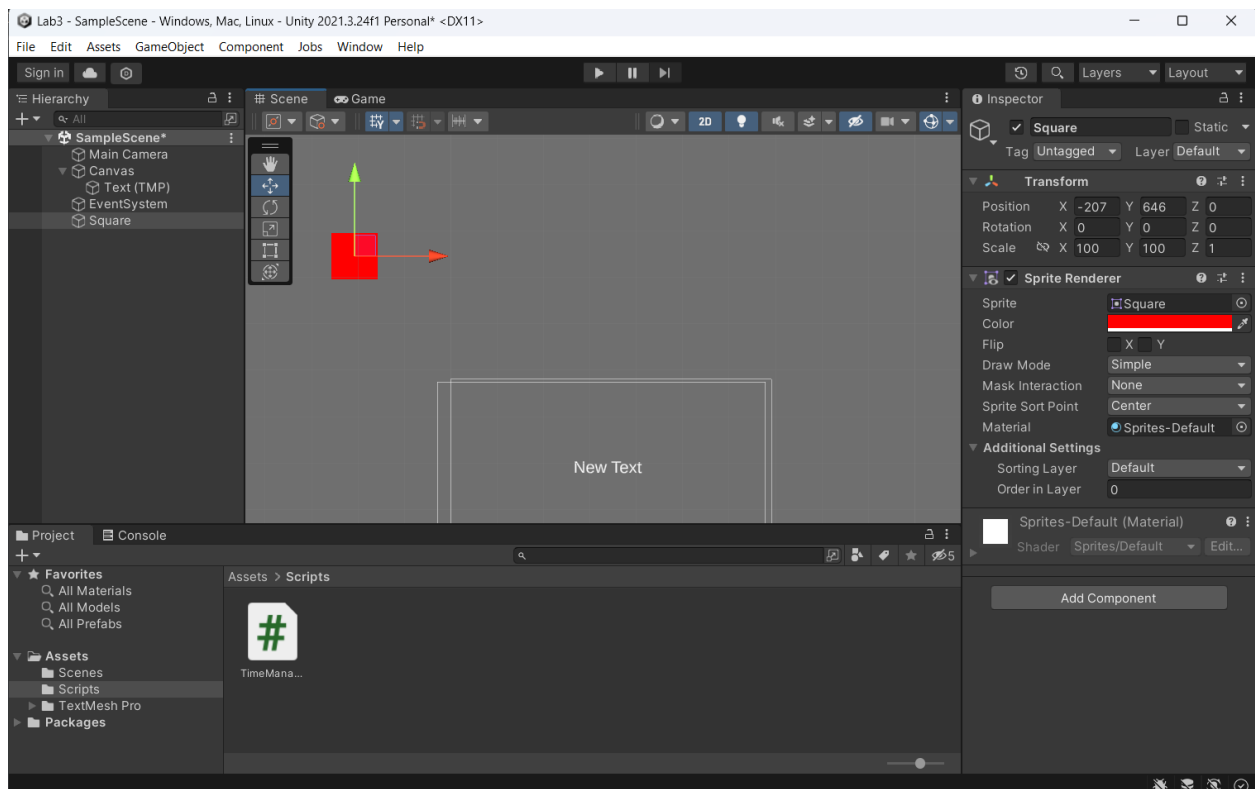
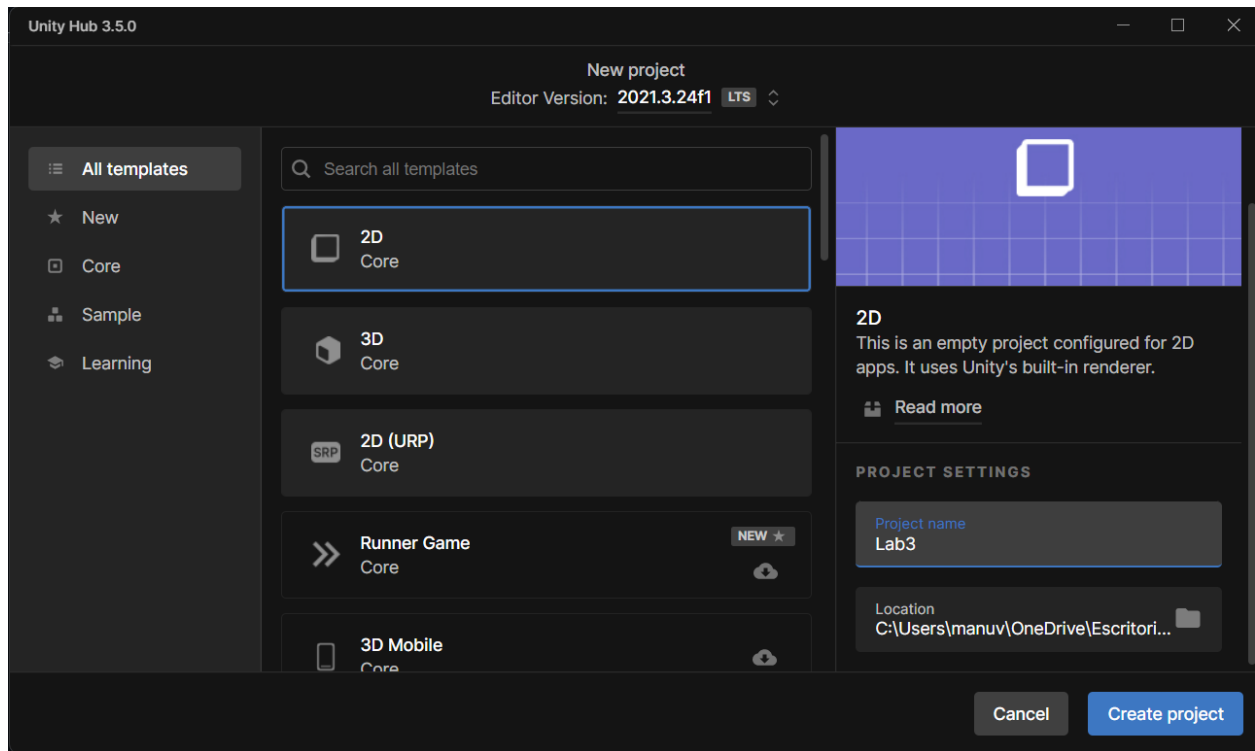
18 de agosto del 2023

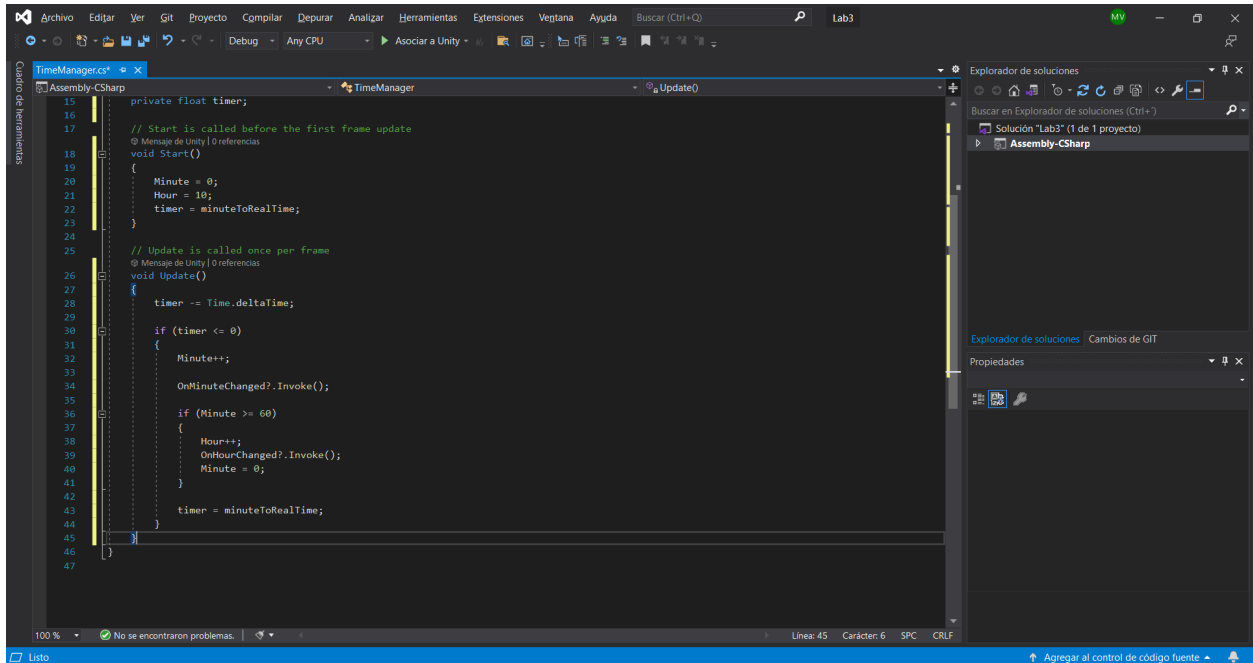
Modelación de sistemas multiagentes con gráficas computacionales
(Gpo 102)

Denisse Lizbeth Maldonado Flores

Alejandro Fernández

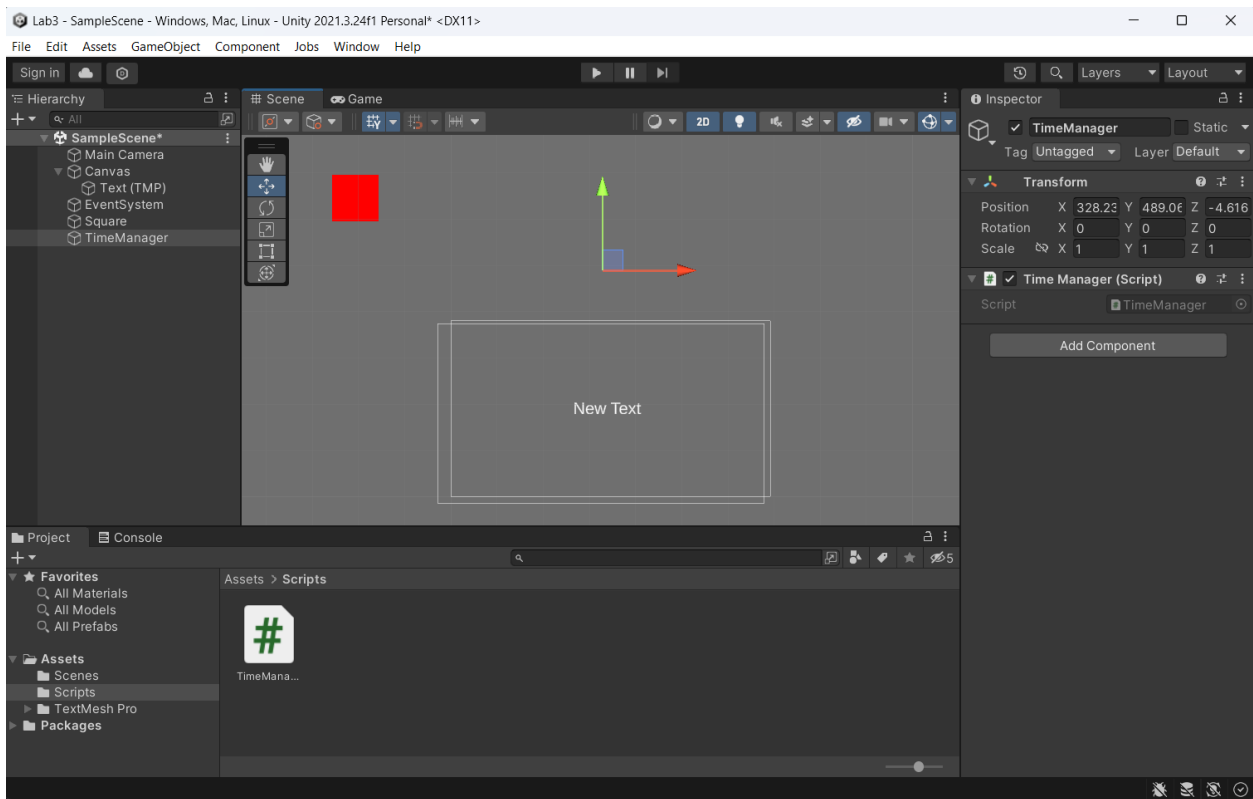
- Screenshots de los 9 pasos realizados.





The screenshot shows the Visual Studio Code editor with the `TimeManager.cs` script open. The script is written in C# and implements a timer system. It includes a `Start()` method that initializes the timer and an `Update()` method that updates the timer every frame. The timer is a private float variable named `timer`. The `Start()` method sets `Minute` to 0, `Hour` to 10, and `timer` to `minuteToRealTime`. The `Update()` method decrements `timer` by `Time.deltaTime` and checks if it is less than or equal to 0. If so, it increments `Minute` and calls `OnMinuteChanged?.Invoke()`. If `Minute` is greater than or equal to 60, it increments `Hour` and calls `OnHourChanged?.Invoke()`, then resets `Minute` to 0. Finally, it sets `timer` to `minuteToRealTime`.

```
15 private float timer;
16
17 // Start is called before the first frame update
18 @ Message de Unity | 0 referencias
19 void Start()
20 {
21     Minute = 0;
22     Hour = 10;
23     timer = minuteToRealTime;
24 }
25
26 // Update is called once per frame
27 @ Message de Unity | 0 referencias
28 void Update()
29 {
30     timer -= Time.deltaTime;
31
32     if (timer <= 0)
33     {
34         Minute++;
35         OnMinuteChanged?.Invoke();
36
37         if (Minute >= 60)
38         {
39             Hour++;
40             OnHourChanged?.Invoke();
41             Minute = 0;
42         }
43         timer = minuteToRealTime;
44     }
45 }
46
47
```



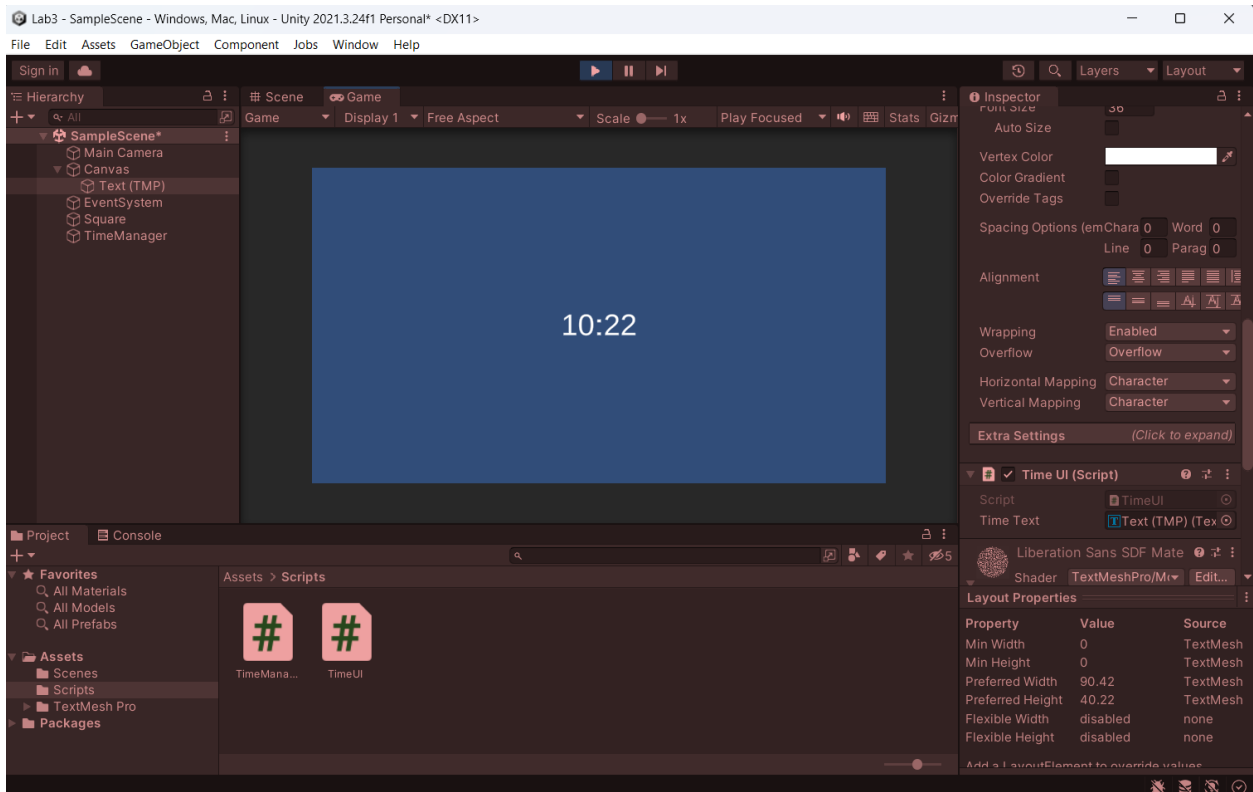
```

Archivo  Editar  Ver  Git  Proyecto  Compilar  Depurar  Analizar  Herramientas  Extensiones  Ventana  Ayuda  Buscar (Ctrl+Q)  Lab3

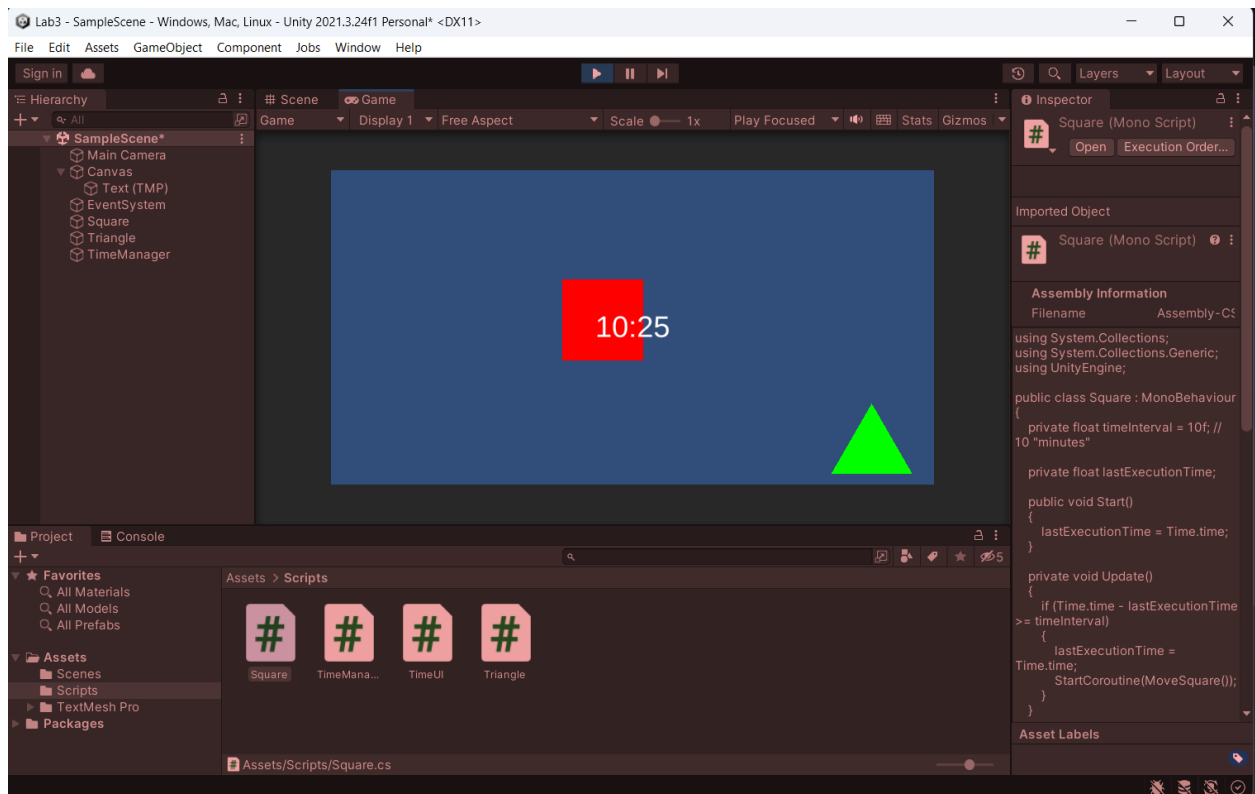
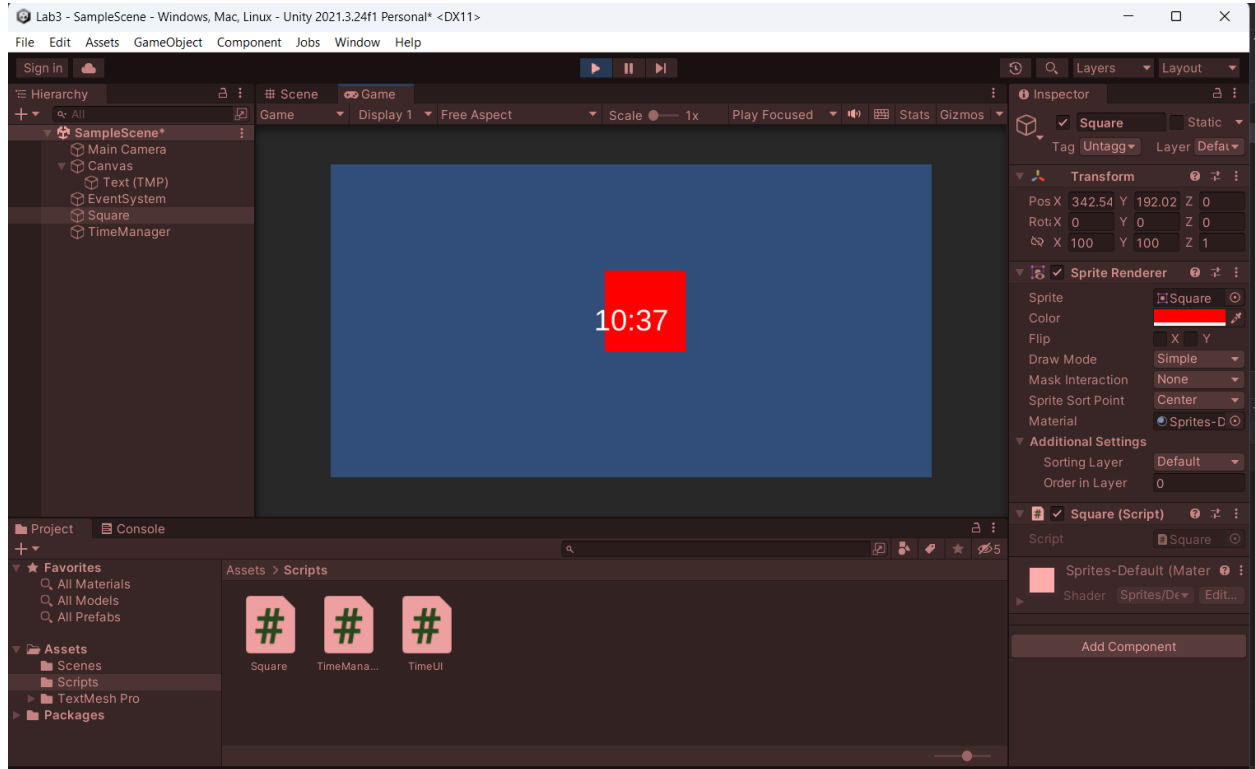
TimeUI.cs  TimeManager.cs
Assembly-CSharp  NewBehaviourScript  UpdateTime()

1  using System.Collections;
2  using System.Collections.Generic;
3  using UnityEngine;
4  using TMPro;
5
6  public class NewBehaviourScript : MonoBehaviour
7  {
8
9      public TextMeshProUGUI timeText;
10
11      // Start is called before the first frame update
12      private void OnEnable()
13      {
14
15          TimeManager.OnMinuteChanged += UpdateTime;
16          TimeManager.OnHourChanged += UpdateTime;
17      }
18
19      // Update is called once per frame
20      private void OnDisable()
21      {
22
23          TimeManager.OnMinuteChanged -= UpdateTime;
24          TimeManager.OnHourChanged -= UpdateTime;
25      }
26
27      private void UpdateTime()
28      {
29          timeText.text = $"{TimeManager.Hour.ToString("00")}:{TimeManager.Minute:00}";
30      }
31
32
33
34
100 %  No se encontraron problemas.  Linea: 28  Carácter: 5  SPC  CRLF

```



```
11 }
12
13 Mensaje de Unity | 0 referencias
14 public void OnDisable()
15 {
16     TimeManager.OnMinuteChanged -= TimeCheck;
17 }
18
19 2 referencias
20 private void TimeCheck()
21 {
22     if (TimeManager.Hour == 10 && TimeManager.Minute == 30)
23     {
24         StartCoroutine(MoveSquare());
25     }
26 }
27
28 1 referencia
29 private IEnumerator MoveSquare()
30 {
31     transform.position = new Vector3(-207f, 646f, 0);
32     Vector3 targetPos = new Vector3(345.8f, 190.2f, 0);
33
34     Vector3 currentPos = transform.position;
35
36     float timeElapsed = 0;
37     float timeToMove = 3;
38
39     while (timeElapsed < timeToMove)
40     {
41         transform.position = Vector3.Lerp(currentPos, targetPos, timeElapsed / timeToMove);
42         timeElapsed += Time.deltaTime;
43         yield return null;
44     }
45 }
46
```



- **Link de video con una duración no mayor a 30 segundos del resultado final del laboratorio.**

<https://youtu.be/aPDSP9-KQpU>

- **Reflexión final sobre el aprendizaje obtenido en este laboratorio así como un listado de al menos 5 ideas que identifiques donde puedes aplicar este concepto en tu proyecto.**

En este laboratorio, aprendí a usar la librería del sistema en Unity para gestionar el tiempo y crear eventos basados en intervalos de tiempo. Esto es fundamental para crear mecánicas de juego dependientes del tiempo y para sincronizar acciones.

Ideas para aplicación en algun juego real:

- **Sistema de Misiones:** Activar misiones automáticamente en intervalos de tiempo para mantener el juego interesante.
- **Crecimiento de Recursos:** Hacer que los recursos se regeneren gradualmente con el tiempo para mantener la jugabilidad constante.
- **Eventos del Mundo:** Introducir eventos diurnos/nocturnos que cambien el juego y desafíos para los jugadores.
- **Horarios de NPCs:** Hacer que los NPCs estén disponibles para interactuar en momentos específicos, agregando realismo.
- **Economía Simulada:** Cambiar los precios de los objetos en función de la oferta y la demanda simulada para enriquecer la economía del juego.